(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号

特表2004-531878

(P2004-531878A)

(43)公表日　平成16年10月14日(2004.10.14)

(51)Int.Cl.[7]	F I	テーマコード（参考）
H01L 21/02	H01L 21/02 Z	

審査請求 未請求　予備審査請求 有　（全 42 頁）

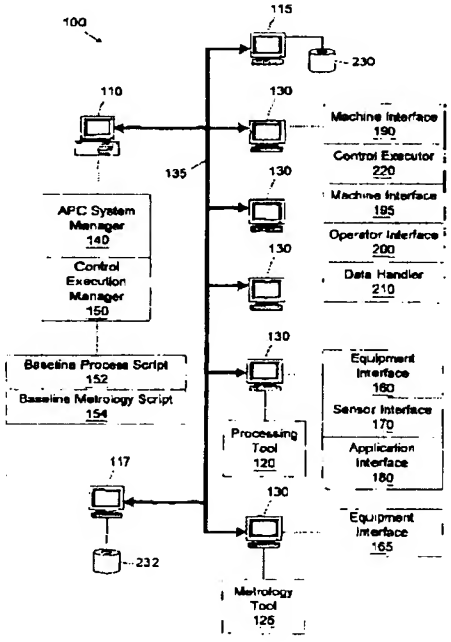
(21)出願番号　特願2002-568120(P2002-568120)
(86)(22)出願日　平成13年10月22日(2001.10.22)
(85)翻訳文提出日　平成15年8月21日(2003.8.21)
(86)国際出願番号　PCT/US2001/050178
(87)国際公開番号　WO2002/069063
(87)国際公開日　平成14年9月6日(2002.9.6)
(31)優先権主張番号　09/789,871
(32)優先日　平成13年2月21日(2001.2.21)
(33)優先権主張国　米国(US)

(71)出願人　591016172
アドバンスト・マイクロ・ディバイシズ・インコーポレイテッド
ADVANCED MICRO DEVICES INCORPORATED
アメリカ合衆国、９４０８８－３４５３　カリフォルニア州、サニィベイル、ピィ・オゥ・ボックス・３４５３、ワン・エイ・エム・ディ・プレイス、メイル・ストップ・６８（番地なし）
(74)代理人　100099324
弁理士　鈴木　正剛
(74)代理人　100111615
弁理士　佐野　良太

最終頁に続く

(54)【発明の名称】ベースライン制御スクリプトを用いてツールを制御するための方法および装置

(57)【要約】
製造システム（１００）を制御するための方法が、複数のツール（１２０、１２５）で被加工物を処理し、この複数のツール（１２０、１２５）のうちの選択したツール（１２０、１２５）についてベースライン制御スクリプト（１５２、１５４）を開始し、ベースライン制御スクリプト（１５２、１５４）のコンテキスト情報を提供し、コンテキスト情報に基づいてツールのタイプを判定し、選択したツール（１２０、１２５）の制御ルーチンをツールのタイプに基づいて選択し、制御ルーチンを実行して選択したツール（１２０、１２５）のコントロールアクションを生成することを含む。製造システム（１００）は、被加工物を処理するように作られた複数のツール（１２０、１２５）と、制御実行マネージャ（１５０）と、コントロールエグゼキュータ（２２０）とを含む。制御実行マネージャ（１５０）は、複数のツール（１２０、１２５）のうちの選択したツール（１２０、１２５）についてベースライン制御スクリプト（１５２、１５４）を開始し、ベースライン制御スクリプト（１５２、１５４）のコンテキスト情報を提供するように作ら



【特許請求の範囲】
【請求項１】
複数のツール（１２０、１２５）で被加工物を処理する際に、
前記複数のツール（１２０、１２５）のうちの選択したツール（１２０、１２５）についてベースライン制御スクリプト（１５２、１５４）を開始する処理と、
前記ベースライン制御スクリプト（１５２、１５４）のコンテキスト情報を提供する処理と、
前記コンテキスト情報に基づいてツールのタイプを判定する処理と、
前記選択したツール（１２０、１２５）の制御ルーチンをツールのタイプに基づいて選択する処理と、
前記制御ルーチンを実行して前記選択したツール（１２０、１２５）のコントロールアクションを生成する処理とを含む、製造システム（１００）を制御するための方法。
【請求項２】
前記制御ルーチンを選択する処理が、制御ルーチンのライブラリ（２４０）にリンクする処理をさらに含む、請求項１に記載の方法。
【請求項３】
前記コンテキスト情報が前記選択したツールに関連したエンティティ識別コードを含み、前記ツールのタイプを判定する処理が前記エンティティ識別コードに基づいてツールのタイプを判定する処理をさらに含む、請求項１に記載の方法。
【請求項４】
前記コンテキスト情報がオペレーション識別コードを含み、前記制御ルーチンを選択する処理が前記ツールのタイプと前記オペレーション識別コードとに基づいて制御ルーチンを選択する処理をさらに含む、請求項１または３に記載の方法。
【請求項５】
前記コンテキスト情報が製品識別コードを含み、前記制御ルーチンを選択する処理がツールのタイプと前記製品識別コードとに基づいて制御ルーチンを選択する処理をさらに含む、請求項１、３または４に記載の方法。
【請求項６】
被加工物を処理するように作られた複数のツール（１２０、１２５）と、
前記複数のツール（１２０、１２５）のうちの選択したツール（１２０、１２５）についてベースライン制御スクリプト（１５２、１５４）を開始し、前記ベースライン制御スクリプト（１５２、１５４）のコンテキスト情報を提供するように作られた制御実行マネージャ（１５０）と、
前記ベースライン制御スクリプト（１５２、１５４）を実行し、前記コンテキスト情報に基づいてツールのタイプを判定し、前記選択したツール（１２０、１２５）の制御ルーチンをツールのタイプに基づいて選択し、前記制御ルーチンを実行して前記選択したツール（１２０、１２５）のコントロールアクションを生成するように作られたコントロールエグゼキュータ（２２０）とを含む、製造システム（１００）。
【請求項７】
前記コントロールエグゼキュータ（２２０）が制御ルーチンのライブラリ（２４０）にリンクして制御ルーチンを選択するように作られている、請求項６に記載のシステム（１００）。
【請求項８】
前記コンテキスト情報が前記選択したツールに関連したエンティティ識別コードを含み、コントロールエグゼキュータ（２２０）が前記エンティティ識別コードに基づいてツールのタイプを判定するように作られている、請求項６に記載のシステム（１００）。
【請求項９】
前記コンテキスト情報がオペレーション識別コードを含み、前記コントロールエグゼキュータ（２２０）がツールのタイプと前記オペレーション識別コードとに基づいて制御ルーチンを選択するように作られている、請求項６または８に記載のシステム（１００）。

【請求項１０】
前記コンテキスト情報が製品識別コードを含み、前記コントロールエグゼキュータ（２２０）がツールのタイプと前記製品識別コードとに基づいて制御ルーチンを選択するように作られている、請求項６、８または９に記載のシステム（１００）。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は広義には半導体デバイス製造の分野に関し、特に、ベースライン制御スクリプトを用いてツールを制御するための方法および装置に関する。
【背景技術】
【０００２】
半導体業界では、マイクロプロセッサ、メモリデバイスなどの集積回路デバイスの品質、信頼性、スループットを向上させたいという強い要求が常にある。この要求は、それまでよりも確実に動作する高品質のコンピュータや電子デバイスを求める消費者需要を受けてなお一層高まっている。こうした需要があることから、トランジスタなどの半導体デバイスの製造ならびにこのようなトランジスタを組み入れた集積回路デバイスの製造の点でたゆみない改善がなされてきている。また、典型的なトランジスタの構成部品の製造時に欠陥が少なくなることで、トランジスタ１つあたりの総コストの削減になるほか、このようなトランジスタを組み入れた集積回路デバイスのコストの削減にもなる。
【０００３】
通常、フォトリソグラフィ用ステッパ、エッチツール、成膜ツール、研磨ツール、高速熱処理ツール、インプランテーションツールなどを含むさまざまな処理ツールを利用して、一連の処理ステップが多数のウエハ上で実施される。ここ数年、半導体処理ツールの基礎をなすいくつかの技術が次第に注目されるようになり、相当な改良がなされている。しかしながら、この分野でこうした進歩があったにもかかわらず、現段階で商用入手可能な処理ツールの多くにはある種の欠点がある。特に、このようなツールには、ユーザが利用しやすい形式でパラメータデータの履歴を提供する機能や、イベントログの記録、最新の処理パラメータとラン（実行）全体の処理パラメータのリアルタイムでのグラフィック表示、遠隔地すなわちローカルサイトおよび世界中で監視を行う機能などの高度なプロセスデータ監視能力が欠けていることが多い。これらの欠点が原因で、スループットや精度、安定性、再現性などの重要な処理パラメータ、処理温度、機械的なツールのパラメータなどが最適とはいえない形で制御される結果になりかねない。この多様性はラン内での格差、ランとランとの格差、ツールとツールとの格差として表れ、こうした格差が進むと製品の品質や性能の偏りへとつながる可能性があるのに対し、これらのツールのための理想的な監視・診断システムがあれば重要なパラメータの制御を最適化するための手段が得られるだけでなく上記の多様性を監視する手段も得られるであろう。
【０００４】
半導体処理ラインのオペレーションを改善するためのひとつの手法として全工場規模の制御システムを採用してさまざまな処理ツールのオペレーションを自動制御することがあげられる。製造ツールが製造フレームワークと通信または処理モジュールのネットワークと通信する。それぞれの製造ツールは一般に機器インタフェースと接続されている。機器インタフェースは、製造ツールと製造フレームワークとの間の通信をしやすくするマシンインタフェースと接続されている。マシンインタフェースは通常、高度プロセス制御（ＡＰＣ）システムの一部であってもよいものである。ＡＰＣシステムは製造モデルに基づいて制御スクリプトを開始するが、この制御スクリプトは製造プロセスを実行するのに必要なデータを自動的に取り込むソフトウェアプログラムであってもよい。しばしば半導体デバイスは複数の製造ツールによって複数のプロセスで段階をおって作られ、処理後の半導体デバイスの品質に関係するデータが生成される。
【０００５】
製作プロセスの間、製作対象となるデバイスの性能に影響するさまざまな出来事が起こる

可能性がある。つまり、製作プロセスステップのばらつきが原因でデバイス性能のばらつきが発生する。構造の臨界寸法、ドーピングレベル、接触抵抗、粒子汚染などの要因がすべてデバイスの最終性能に影響する可能性を持つことがある。処理ラインのツールはそれぞれ処理のばらつきを減らすべく性能モデルに従って制御されている。普通に制御されるツールとしては、フォトリソグラフィ用ステッパ、研磨ツール、エッチングツール、成膜ツールがあげられる。これらのツール用のプロセスコントローラには前処理および／または後処理測定データが供給される。処理時間などの動作レシピパラメータについては、性能モデルと計測情報をもとにプロセスコントローラで計算し、標的値にできるだけ近い処理結果を達成しようと試みられている。このようにしてばらつきを減らすことで、スループットを向上させ、コストを削減し、デバイス性能をそれまでよりも高くできるなどの結果につながるが、これらはいずれも収益性の向上に結び付く。

【発明の開示】

【発明が解決しようとする課題】

【０００６】

コンフィギュレーション制御と効率の問題は、全工場規模のＡＰＣシステムなどの分散コンピューティング環境では普通に見られるものである。一般に、プロセスコントローラを構築するための制御コードは多くのソフトウェア開発者によって書かれている。個々の開発者が特定タイプのコントローラを開発しつつ広範囲にわたって作業をすることもある。彼ら開発者がそれぞれ独自のプログラミングスタイルを持ち、自分で作り出したルーチンに頼っていることは珍しくない。たとえば、ＡＰＣフレームワーク内のデータベースや他のエンティティと接続し、さまざまな数学関数や基本のユーティリティ関数を実行するためのルーチンのセットを開発者ごとに持っている場合がある。

【０００７】

このような形態をとることに伴うひとつの問題として、プロセス制御スクリプト間の一貫性がほとんどない点があげられる。開発者独自のスクリプトが多数あることもコンフィギュレーション制御の問題や効率面での問題の原因となる。開発者らは、すでに別の開発者が違うタイプのプロセスコントローラ向けに開発したであろうものと同様のコードを書くのに相当な時間を費やす場合がある。標準から外れたコードのデバッグにもさらに多くの時間を要し、余計に効率が悪くなる。

【０００８】

本発明は、上述した問題のうちひとつまたはそれ以上の影響を解決するか少なくとも低減しようとするものである。

【課題を解決するための手段】

【０００９】

発明の開示

本発明の一態様は、製造システムを制御するための方法に見られる。この方法は、複数のツールで被加工物を処理し、この複数のツールのうちの選択したツールについてベースライン制御スクリプトを開始し、ベースライン制御スクリプトのコンテキスト情報を提供し、このコンテキスト情報に基づいてツールのタイプを判定し、選択したツールの制御ルーチンをツールのタイプに基づいて選択し、制御ルーチンを実行して選択したツールのコントロールアクションを生成することを含む。

【００１０】

本発明のもうひとつの態様は、被加工物を処理するように作られた複数のツールと、制御実行マネージャと、コントロールエグゼキュータとを含む製造システムに見られる。制御実行マネージャは、複数のツールのうちの選択したツールについてベースライン制御スクリプトを開始し、ベースライン制御スクリプトのコンテキスト情報を提供するように作られている。コントロールエグゼキュータは、ベースライン制御スクリプトを実行し、このコンテキスト情報に基づいてツールのタイプを判定し、選択したツールの制御ルーチンをツールのタイプに基づいて選択し、制御ルーチンを実行して選択したツールのコントロールアクションを生成するように作られている。

【００１１】
添付の図面を参照して以下の説明を参照することで本発明を理解することができよう。図中、同様の構成要素には同様の参照符号を付してある。
【００１２】
本発明はさまざまな形に改変および変更が可能なものであるが、一例としてその特定の実施形態を図示し、本願明細書中にて詳細に説明してある。しかしながら、特定の実施形態に関する本願明細書の記載がここに開示の特定の形態に本発明を限定することを意図したものではなく、添付の特許請求の範囲に規定の本発明の範囲に含まれる改変例、等価物および代わりとなる物をすべて包含することを意図している点を理解されたい。
【発明を実施するための最良の形態】
【００１３】
発明の実施の形態
以下、本発明の実施形態について説明する。明瞭な説明とするために、本願明細書では実際に使われる実装の構造すべてについて説明してあるとは限らない。もちろん、このような実際の実施形態を開発するにあたって、実装ごとに異なるシステム関連の制約およびビジネス関連の制約に従うなど、実装ごとに固有の多数の決定を行い、開発者ごとに異なる目標を達成しなければならない点は理解できよう。さらに、このような開発の取り組みが複雑で時間を要するものとなる可能性があるとはいえ、本願開示の利益を享受する当業者の日常作業になろうものであることも理解できよう。
【００１４】
以下、図面を参照するが、まず図１を参照すると、高度プロセス制御（ＡＰＣ）システム１００の簡略ブロック図が示されている。ＡＰＣシステム１００は、ランとランとの間の制御や故障の検出／分類を可能にしている取り替え可能な標準化ソフトウェアコンポーネントからなる分散ソフトウェアシステムである。これらのソフトウェアコンポーネントは、半導体製造装置・材料に関する業界団体（ＳＥＭＩ）コンピュータ統合生産（ＣＩＭ）フレームワーク対応のシステム技術仕様および高度プロセス制御（ＡＰＣ）フレームワークに基づくアーキテクチャ標準を満たしている。ＣＩＭ（ＣＩＭフレームワークドメインアーキテクチャに関するＳＥＭＩ　Ｅ８１－０６９９－暫定仕様書）およびＡＰＣ（ＣＩＭフレームワーク高度プロセス制御コンポーネントに関するＳＥＭＩ　Ｅ９３－０９９９－暫定仕様書）の各仕様についてはＳＥＭＩから一般に入手可能である。この特有のアーキテクチャは、オブジェクト指向プログラミングを利用しているソフトウェアに極めて大きく依存し、分散オブジェクトシステム向けにオブジェクトマネジメントグループ（ＯＭＧ）が策定した共通オブジェクトリクエストブローカーアーキテクチャ（ＣＯＲＢＡ）およびＣＯＲＢＡサービスの仕様を採用している。このＯＭＧ　ＣＯＲＢＡアーキテクチャについての情報および仕様は容易に一般に入手可能である。本願明細書にて説明するようなＡＰＣシステム１００の機能を果たすようにすることのできるソフトウェアシステムの一例として、ケーエルエー・テンコール・インコーポレイテッド（ＫＬＡ－Ｔｅｎｃｏｒ，Ｉｎｃ．）から提供されているＣａｔａｌｙｓｔシステムがあげられる。
【００１５】
これらのコンポーネントはＣＯＲＢＡインタフェース定義言語（ＩＤＬ）を使って互いに通信し、その対話をサポートするための共通のサービスセットに頼っている。分散オブジェクトサービスの標準セットはＯＭＧによって規定されている。これらのサービスには以下のものがある。
【００１６】
ＣＯＲＢＡ－コンポーネントとコンポーネントとの間での直接対話に必ず使われる標準ベースの通信プロトコルである。オブジェクト指向の遠隔呼出し通信モデルに従って標準インタフェースを定義することができる。これらのインタフェースとすべてのＡＰＣ通信はＩＤＬを使って定義される。コンポーネントは互いに他のインタフェースでオペレーションを呼び出すことによって通信する。コンポーネント間ではオペレーションパラメータおよび返り値としてデータの受け渡しがなされる。

【００１７】
ＯＭＧイベントサービス－コンポーネント間での非同期的な通信をサポートするものである。ＡＰＣオブジェクトの多くは状態が変化したときにイベントを発信する。これらのイベントは、該当するイベントサブスクライバによって受信される。ＡＰＣシステム内でのイベント利用の例としては、通信コンポーネントの状態（エラー状態を含む）、故障の検出および分類ソフトウェアで検出した故障アラームの通知、マシンのステータスと収集したデータの報告があげられるが、これに限定されるものではない。
【００１８】
ＯＭＧトレーディングサービス－コンポーネントが自己と対話すべき別のコンポーネントを見つけることができるようにするものである。コンポーネントをインストールすると、そのサービスの説明（サービスのオファー）がトレーディングサービスにエクスポートされる。別のコンポーネントが特定の規準を満たすサービスプロバイダの一覧を後から要求することができる。トレーディングサービスでは、要求されたサービスを提供できる他のコンポーネントの一覧を提供する。この機能は、コンポーネントの起動時にひとつのコンポーネントが自己と対話しなければならない他のコンポーネントを見つけることができるようにする目的で使われる。また、これはプラン実行コンポーネントがプランに指定された必要な能力を提供するのにケイパビリティプロバイダ（Ｃａｐａｂｉｌｉｔｙ　Ｐｒｏｖｉｄｅｒ）を見つけなければならないプランスタートアップ（Ｐｌａｎ　Ｓｔａｒｔｕｐ）時にも使われる。
【００１９】
これらのサービスは従来技術において周知である。ＯＭＧのＣＯＲＢＡ／ＩＩＯＰ仕様の文書およびＣＯＲＢＡサービス仕様の文書は当業者に広く配布されており、そこにはさらに詳細な説明がなされている。
【００２０】
図示の実施形態では、ＡＰＣシステム１００は半導体製造環境を制御できるように作られている。複数のコンポーネントがＣＯＲＢＡインタフェース定義言語（ＩＤＬ）を使って互いに通信する。連携しているソフトウェアコンポーネントは、プロセス制御プラン／ストラテジーを管理し、プロセス機器、計測ツール、アドオンセンサからデータを収集し、この情報を利用してさまざまなプロセス制御アプリケーション／アルゴリズムを呼び出し、プロセスモデルを更新し、該当する場合はツール動作レシピパラメータを修正／ダウンロードする。ＡＰＣシステム１００は半導体生産プロセスを制御するための全工場規模のソフトウェアシステムであるが、これは本発明を実施するにあたって必要なことではない。本発明が教示するストラテジーは、任意の規模で異なるコンピュータシステムに適用可能なものである。
【００２１】
代表的な実装例では、ＡＰＣシステム１００は、ＡＰＣホストコンピュータ１１０と、データベースサーバ１１５、１１７と、処理ツール１２０と、計測ツール１２５と、１つまたはそれ以上のワークステーション１３０とを含む。ＡＰＣシステムの各コンポーネントはバス１３５を介して相互に接続されている。バス１３５は実際は複数の層で構成されて複数のプロトコルを利用するものであってもよい。ＡＰＣシステム１００の全体としてのオペレーションはＡＰＣホストコンピュータ１１０上に常駐するＡＰＣシステムマネージャ１４０によって指示されている。ＡＰＣシステムマネージャ１４０は、ＡＰＣフレームワーク用に開発されたすべてのサーバの管理サービス、コンフィギュレーションサービス、イベントサービス、状態サービス；ＡＰＣシステム１００内のコンポーネントの定義、グルーピング、インストール、マネジメント；診断および監視の目的でアクティビティおよびトレース情報を捕捉するための中央サービス；セットアップ値、システム環境設定を含むコンポーネントコンフィギュレーション情報の中央の収納場所；従属オブジェクトおよびイベントチャネルのリストを提供するものである。しかしながら、別の実施形態では、これらの機能を、ベースマネージャ、システムマネージャ、ロガー、レジストリなどの１つまたはそれ以上のソフトウェアコンポーネントに分割してもよい。

【００２２】
ＡＰＣシステム１００は複数の処理モジュールからなるネットワークを含む。これらの処理モジュールは「インテグレーションコンポーネント」と呼ばれることもある。インテグレーションコンポーネントは、既存のファクトリーシステムとのインタフェースとして機能し、ＡＰＣプランを走らせることができるようにするものである。「ＡＰＣプラン」とは、詳細については後述するように特定のタスクを実行するために呼ばれるアプリケーションプログラムのことである。インテグレーションコンポーネントについては、ＡＰＣシステム１００内のさまざまな処理資源によって受け入れられる場合があるものとして示してある。これらの特定のホスティングロケーションは例示の目的で提供されている。処理リソース同士は相互に接続されており、システムの複雑さに応じてさまざまなソフトウェアコンポーネントをさまざまなコンピュータに分散させてもよいし、中央に集中させてもよい。この特定の実施形態におけるインテグレーションコンポーネントは各々ソフトウェア的に実装されている。これらのコンポーネントは従来技術において周知のオブジェクト指向プログラミングの手法を使ってＣ＋＋でプログラムされている。ＡＰＣシステム１００の利点のひとつにモジュール構造であるという点があるが、これによってソフトウェアコンポーネントの移植性が得られる。インテグレーションコンポーネントは、ＡＰＣシステムマネージャ１４０、制御実行マネージャ１５０、ツール１２０、１２５に関連した機器インタフェース１６０、１６５、処理ツール１２０に関連したセンサインタフェース１７０、アプリケーションインタフェース１８０、マシンインタフェース１９０、１９５、オペレータインタフェース２００、データハンドラ２１０を含むがこれに限定されるものではない。

【００２３】
制御実行マネージャ１５０は、ＡＰＣシステム１００のオペレーションの「振り付け」を主に担うコンポーネントである。制御実行マネージャ１５０はＡＰＣプランを解釈し、メインスクリプトおよびサブスクリプトを実行し、イベントの命令時にイベントスクリプトを呼び出す。多様なプラン、スクリプト、サブスクリプトをさまざまな実装に用いることができる。さまざまなプラン、スクリプト、サブスクリプトの具体的な数と機能はそれぞれの実装内容に依存する。たとえば、本実施形態は、
データ収集プラン－特定の処理機器からどのデータを収集すべきか、そのデータをどのようにして折り返し報告するかという要件を定義するセンサおよびマシンインタフェースで用いられるデータ構造
期間プラン－データ収集の開始、データ収集の終了など、センサを作動させるトリガー条件とトリガー遅延とを定義するプラン
レポーティングプラン－収集したデータをどのようにすべきかとデータの可用性をいつ信号伝達すべきかを定義するプラン
サンプリングプラン－外部センサによってデータを収集する頻度を定義するプラン
制御プラン－ＡＰＣアクティビティを実行するために一緒に使われるよう設計された制御スクリプトの集合
制御スクリプト－定義された特定の状況下でＡＰＣシステムが実行する一連のアクション／アクティビティ
などのプランを含むがこれに限定されるものではない。

【００２４】
制御実行マネージャ１５０は、処理ツール１２０などの特定のツールについて、すべてのインテグレーションコンポーネント内でユーザが定義したプロセス制御プランの実行をコーディネートする。指示があると、制御実行マネージャ１５０はプランとこれに関連したスクリプトとを読み出す。次に、サブスクリプトを前処理してメインスクリプトおよびイベントスクリプトへのルーチンを提供する。また、プランに指定されている、プランを実行するのに必要な能力の一覧を取得し、必要な能力が得られるしかるべきインテグレーションコンポーネントに接続する。

【００２５】

次に、制御実行マネージャ１５０は、このプランを走らせる責任をコントロールエグゼキュータ２２０に委ねる。図示の実施形態では、制御実行マネージャ１５０はベースライン制御スクリプトを使って実施すべきコントロールアクションを判定する。ベースラインプロセススクリプト１５２は処理ツール１２０などの処理ツールと併用すべきものであり、ベースライン計測スクリプト１５４は計測ツール１２５などの計測ツールと併用すべきものである。ベースラインスクリプト１５２、１５４の詳細については、図２乃至図６を参照して後述する。

【００２６】

制御実行マネージャ１５０は、プランを順次実行してプランの終了またはプラン実行時におけるエラーを制御実行マネージャ１５０に折り返し報告するためのコントロールエグゼキュータ２２０を、適当なベースラインプロセススクリプト１５２またはベースライン計測スクリプト１５４に基づいて作成する。このように、制御実行マネージャ１５０が実行されるすべてのプランの全体としてのマネジメントを担っているのに対し、各コントロールエグゼキュータ２２０はひとつのプランを走らせることだけを担っている。コントロールエグゼキュータ２２０は制御実行マネージャ１５０によって作成され、プランが生きている間は存在し、そのプランが終了または異常終了した旨の報告の後で制御実行マネージャ１５０によって破棄される。制御実行マネージャ１５０は、複数のコントロールエグゼキュータ２２０を介して複数のプランを同時に始めることができる。

【００２７】

マシンインタフェース１９０、１９５は、ＡＰＣシステムマネージャ１４０などのＡＰＣフレームワークと機器インタフェース１６０、１６５との間のギャップを埋めるものである。マシンインタフェース１９０、１９５は、処理ツールまたは計測ツール１２０、１２５とＡＰＣフレームワークとを接続し、マシンのセットアップ、起動、監視、データ収集をサポートする。この特定の実施形態では、マシンインタフェース１９０、１９５は主に機器インタフェース１６０、１６５の特定の通信とＡＰＣフレームワークのＣＯＲＢＡ通信との間の翻訳を行う。特に、マシンインタフェース１９０、１９５は、コマンド、ステータスイベント、収集されたデータを機器インタフェース１６０、１６５から受け取り、必要に応じてこれを他のＡＰＣコンポーネントおよびイベントチャネルに転送する。今度は、他のＡＰＣコンポーネントからの応答がマシンインタフェース１９０、１９５で受信され、機器インタフェース１６０、１６５に送られる。また、マシンインタフェース１９０、１９５は、必要に応じてメッセージとデータを再フォーマットおよび再構築する。マシンインタフェース１９０、１９５は、ＡＰＣシステムマネージャ１４０内でのスタートアップ／シャットダウン手順をサポートする。また、機器インタフェース１６０、１６５によって収集されたデータを一時的に蓄積し、適切なデータ収集イベントを発信するＡＰＣデータコレクタとしても機能する。

【００２８】

センサインタフェース１７０は、処理ツール１２０のオペレーションを監視しているセンサが生成したデータを収集する。センサインタフェース１７０は、ラボビュー（ＬａｂＶＩＥＷ）または他のセンサなどの外部センサやバスをベースにしたデータ取得ソフトウェアと通信するための適当なインタフェース環境を提供するものである。アプリケーションインタフェース１８０は、ラボビュー（ＬａｂＶＩＥＷ）、マセマティカ（Ｍａｔｈｅｍａｔｉｃａ）、モデルウェア（ＭｏｄｅｌＷａｒｅ）、マトラボ（ＭａｔＬａｂ）、シムカ（Ｓｉｍｃａ）４０００、エクセル（Ｅｘｃｅｌ）などの制御プラグインアプリケーションを実行するための適切なインタフェース環境を提供するものである。センサについては相手先商標製品製造（ＯＥＭ）によって処理ツール１２０と一緒に供給してもよいし、ＯＥＭから入手した後でインストールされる「アドオン」センサであってもよい。センサインタフェース１７０はセンサによって生成されたデータを収集する。このセンサは、たとえば動作条件の圧力および温度についてのデータを生成するものであっても構わない。アプリケーションインタフェース１８０はコントロールエグゼキュータ２２０からデータを得て、そのデータに関する計算または分析を実行する。次に、結果をコントロールエグ

ゼキュータ２２０に返す。マシンインタフェース１９０およびセンサインタフェース１７０は、使われるデータを収集するための共通の機能一式を利用する。機器インタフェース１６０は処理ツール１２０についてセンサが収集したそれぞれのデータを集め、集めたデータをマシンインタフェース１９０に送信する。

【００２９】

オペレータインタフェース２００は、グラフィカルユーザインタフェース（ＧＵＩ）（図示せず）によってウエハ製作技術者とＡＰＣシステム１００との通信を容易にするものである。このＧＵＩはＷｉｎｄｏｗｓ（Ｒ）またはＵＮＩＸベースのオペレーティングシステムであっても構わないが、これは本発明を実施するにあたって必要なことではない。特に、いくつかの別の実施形態ではＧＵＩを採用することすらせず、ディスクオペレーティングシステム（ＤＯＳ）ベースのオペレーティングシステムを使って通信を行うことができる。オペレータインタフェース２００はダイアログボックスを表示して情報を提供し、指示を求め、別のデータを収集する。ＣＯＲＢＡインタフェースによって、技術者らはオペレータインタフェース２００のコンポーネントを使って任意の数のディスプレイ群に多様なポップアップダイアログを同時に表示することができる。また、オペレータインタフェース２００は、ポップアップを表示させることのできるディスプレイ群も維持する。さらに、オペレータインタフェース２００は、アナウンスメントオペレーションすなわちメッセージと「ＯＫ」ボタンだけの単純なポップアップを表示する一方的なメッセージを提供する。

【００３０】

データハンドラ２１０は、他のＡＰＣシステム１００のコンポーネントが集めたデータを受け取り、これらのデータをデータベースサーバ１１５、１１７上のデータストア２３０、２３２（リレーショナルデータベースなど）に格納する。データハンドラ２１０は、標準的な構造化照会言語（ＳＱＬ）コマンドを受け取ることのできるものであってもよいし、あるいは、データハンドラ２１０は、異なるタイプのアクセスプロトコルを翻訳してＳＱＬコマンドまたは他の何らかのプロトコルコマンドを生成するものであってもよい。データ格納機能を中央に集めることで、さまざまなコンポーネントの移植性が高くなる。

【００３１】

次に、図２に示す簡略ブロック図を参照してベースライン制御スクリプト１５２、１５４の大まかなオペレーションについて説明する。同図において、ベースライン制御スクリプト１５２、１５４と多様な共有ベースラインライブラリとの間のリンクが示されている。総じて言うと、ベースライン制御スクリプト１５２、１５４は、ＡＰＣシステム１００内の制御スクリプトを開発するためのフレームワークを提供するものである。ベースライン制御スクリプトはライブラリに格納された共有ベースラインコンポーネントを利用する。図示の実施形態では、共有ベースラインコンポーネントは、制御アルゴリズムを定義するための制御ベースラインライブラリ２４０と、共通に使われる数学関数（ｓｕｍ、ｍｅａｎ、ｍｅｄｉａｎなど）を定義するための数学ベースラインライブラリ２５０と、スクリプト実行の通信アスペクト（マシンインタフェース１９５、オペレータインタフェース２００、他のそのような外部コンポーネントを介してのデータストア２３０、２３２、マシンインタフェース１９５、機器インタフェース１６０との対話）を定義するための対話ベースラインライブラリ２６０と、共有の共通関数を定義するためのユーティリティベースラインライブラリ２７０と、施設に特有の他のライブラリ２４０、２５０、２６０、２７０のルーチンに対する例外または関数を定義するための施設ライブラリ２８０と、ベースライン制御スクリプトに対する呼出に含まれる特定のオペレーションＩＤからのレイヤー（例、ポリゲート層）を定義するためのレイヤーライブラリ２９０とを含む。ベースラインライブラリ２４０、２５０、２６０、２７０、２８０、２９０を、ベースライン制御スクリプト１５２、１５４のオペレーションの間にコントロールエグゼキュータ２２０によってリンクさせておいてもよい。

【００３２】

総じて言うと、ベースライン制御スクリプト１５２、１５４は、スクリプトに対する呼出

に含まれる情報と施設ベースラインライブラリ２８０およびレイヤーベースラインライブラリ２９０に含まれる情報とに基づいてコントロールアクションの性質を判定する。ベースライン制御スクリプト１５２、１５４は、制御ベースラインライブラリ２４０とリンクして必要な制御関数にアクセスする。ベースライン制御スクリプト１５２、１５４は対話ベースラインライブラリ２６０とリンクし、コントロールアクションを実施したりツール１２０、１２５の動作レシピの更新目的で機器インタフェース１６０と通信したりするのに使われるデータを集めるための関数にアクセスする。数学ベースラインライブラリ２５０の関数を必要に応じてベースライン制御スクリプト１５２、１５４または他のライブラリの他の関数から呼ぶようにしてもよい。

【００３３】

ここで図３を参照すると、ベースラインプロセススクリプト１５２の構成を示す簡略ブロック図が示されている。ベースラインプロセススクリプト１５２は、アプリケーションコンフィギュレーションブロック３００と、ベースラインアプリケーションセットアップブロック３１０と、コントローラ定数およびコンテキスト固有の設定ブロック３２０と、フィードフォワードデータ分析ブロック３３０と、制御スレッドブロック３４０と、危機（ｊｅｏｐａｒｄｙ）ブロック３５０と、コントロールアクションおよびビジネスルールブロック３６０と、結果ブロック３７０とを含む。

【００３４】

アプリケーションコンフィギュレーションブロック３００内には、機器インタフェース１６０からの呼出に含まれる情報に基づいてコントローラが使うユーザグローバルコンフィギュレーション変数が定義されている。これは、レシピ管理システム（ＲＭＳ）からの変数の値（すなわちレシピ設定のグローバルデータベース）と必要なコンテキスト変数とを含む。コンテキスト変数の値は制御スレッドを定義し、一般にツール識別コード、ロット番号、オペレーション番号などの変数の値で構成されている。また、必要なベースライン変数も一定の値である。一例として、エラー通知用の電子メール一覧、タイムアウトの値、「子」ロットとみなされる１ロットで許容される最大ウエハ数、コントローラが使った前のレイヤ情報（フィードフォワード情報）などがあげられる。

【００３５】

ベースラインアプリケーションセットアップブロック３１０は、アプリケーションコンフィギュレーションブロック３００でのセット時にロット番号とウエハ数量の値を利用し、ロット番号、ファミリ名、親名、施設、ウエハ数、ステータス（すなわちロットが親ロットであるか子ロットであるか）の値を返す。また、ベースラインアプリケーションセットアップブロック３１０は、コントローラからのポップアップウィンドウとすべてのポップアップウィンドウタイトルの最初の部分の送り先である端末のリストを設定する。

【００３６】

コントローラ定数およびコンテキスト固有の設定ブロック３２０はすでに定義されたコンテキストとＲＭＳ情報とを利用して、制御の動きを計算するのにコントローラが使う値を設定する。たとえば、コントローラ定数およびコンテキスト固有の設定ブロック３２０は、ＲＭＳに定義された値に基づいて、コンテキスト情報（または「スレッド」指定）を使って制御モデルパラメータの値を設定することができる。その具体的な一例は、特定のエッチチャンバのコンテキストに基づいて制御モデルで使われるエッチレートの値や、ＲＭＳに定義されているようなエッチチャンバのエッチレートの値を設定することであろう。また、コントローラ定数およびコンテキスト固有の設定ブロック３２０は、アプリケーションコンフィギュレーションブロック３００に設定されているようなロット番号およびレイヤ名ごとのクエリを使ってデータベースからフィードフォワード情報を読み出す。

【００３７】

フィードフォワードデータ分析ブロック３３０は、特定のロットに関連したデータのアレイに含まれるエレメントをチェックし、抜けている値にはデフォルト値を埋める。たとえば、前のプロセスのターゲットを使って、コントローラが使うフィードフォワード情報の一部として必要な抜けている測定値を設定することができる。デフォルト値を使わずに抜

けているフィードフォワード情報の値を設定するための他の方法をフィードフォワードデータ分析ブロック３３０で実施するようにしてもよい。

【００３８】

制御スレッドブロック３４０は、データストア２３０、２３２を照会して現在の制御スレッドに関連した制御状態を読み出すのに必要なキーと状態構造の値を設定する。これらのキーはデータストア２３０、２３２からスレッド状態データを読み出すのに使われる。制御スレッドブロック３４０は、このスレッドコンテキストで処理された新しいロットの順に並べられたデータのスタックの中のスレッド状態データを検索する。このような値が見つかった場合、これは制御モデルを含むユーザ定義関数に渡され、この関数でスレッド状態の値を計算して返す。スタックに値が見つからなかった場合、制御スレッドブロック３４０は階層を上がって検索し、スレッド状態の値のある最初の階層レベルからデータを読み出す。スタックおよびすべての階層レベルは同様ではあるが精度の異なるデータを含むと仮定される。

【００３９】

危機ブロック３５０は、データベース内でルックアップを実行し、危機スタックのロット数（すなわち、最後の計測オペレーション以降に特定のスレッドで処理されたロットのスタック）についての値を読み出す。この値はこの危機カテゴリ内のロット数についての閾値すなわち一般にＲＭＳで指定される値と比較される。この閾値以下である場合、コントローラはそのまま残る。閾値を超えた場合、危機スタックのロットのリストからのひとつのロットについて計測イベントを実施するようオペレータに指示するポップアップディスプレイを表示してコントローラは異常終了する。

【００４０】

コントロールアクションおよびビジネスルールブロック３６０はコントローラの核心である。コントロールアクションおよびビジネスルールブロック３６０は状態およびターゲット情報からコントローラ入力（プロセスレシピ更新）を計算する。これらの結果はグローバル制御結果アレイに置かれる。次に、コントロールアクションおよびビジネスルールブロック３６０はビジネスルールを実行し、コントローラのユーザ入力オーバーライドに従ってプロセスレシピ更新のチェッキングを制限および／またはプロセスレシピ更新を設定する。

【００４１】

結果ブロック３７０は、プロセスレシピ更新、データ計算／フォーマットを含むコントロールアクションおよびビジネスルールブロック３６０からの出力あるいはイベントを受け、これをまとめて一時的に蓄積し、データをフォーマットする。結果ブロック３７０は一時的に蓄積したデータを機器インタフェース１６０に送信し、機器インタフェース１６０に対してマシンインタフェース１９５によってセットアップ／スタートマシン呼出を開始する。次に、結果ブロック３７０は、現在のコンテキスト（スレッド）についてロット番号およびレイヤに対してデータストア２３０、２３２に格納されたデータを格納する。危機スタックも最後の計測イベント以降に処理された追加のロットとして現在のロットで更新される。

【００４２】

図４を参照すると、ベースライン計測スクリプト１５４の構成を示す簡略ブロック図が示されている。ベースライン計測スクリプト１５４は、計測ツールセットアップブロック４００と、アプリケーションコンフィギュレーションブロック４１０と、ベースラインアプリケーションセットアップブロック４２０と、受入れツールデータブロック４３０と、コントローラ定数およびコンテキスト固有の設定ブロック４４０と、制御スレッドブロック４５０と、モデル更新ブロック４６０と、結果ブロック４７０とを含む。

【００４３】

計測ツールセットアップブロック４００内では、データ収集を開始して一時的に蓄積されていたデータをコントロールエグゼキュータ２２０に送信する。機器インタフェース１６５へのマシン呼出をセットアップ／開始するためにマシンインタフェース１９０も開始さ

れる。

【００４４】

アプリケーションコンフィギュレーションブロック４１０およびベースラインアプリケーションセットアップブロック４２０は、ベースラインプロセススクリプト１５２に関連して上述した同じ名前のブロックと同様の機能を果たす。

【００４５】

受入れツールデータブロック４３０は、ベースライン計測スクリプト１５４を一時停止し、一般に計測ツールであるデータソースからのデータセットを待つ。このイベントの待ち時間とスクリプトの一時停止を解除するイベントの名前が受入れツールデータブロック４３０に指定されている。

【００４６】

コントローラ定数およびコンテキスト固有の設定ブロック４４０は、ベースラインプロセススクリプト１５２に関連して上述した同じ名前のブロックと同様の機能も果たす。

【００４７】

制御スレッドブロック４５０は、現在のスレッドについて算出した制御状態をデータストア２３０、２３２に格納するのに必要なキーおよび状態構造の値を設定する。また、制御スレッドブロック４５０は、スレッド状態を更新するのに必要なすべての値を計算する。この関数は、定義されたグローバル変数を読み取り、必要とされる結果を計算する。これらの結果には、ロット平均、プロセスレート、ターゲットまたは予測からの偏差など、コントローラの更新時に使用される統計値または値が含まれる。この関数の結果はグローバル制御結果アレイにおかれる。

【００４８】

モデル更新ブロック４６０は、ビジネスルール、スペックリミットチェッキング、コントローラのオーバーライドを実施するのに使われる。この関数は、定義されたグローバル変数を読み取り、最終結果を設定する。モデル更新ブロック４６０は、コントローラを更新するのに使用される値ならびに制御履歴に記録される値の設定を担っている。この関数の結果はグローバル制御結果アレイにおかれる。

【００４９】

結果ブロック４７０は、コントローラ定数およびコンテキスト固有の設定ブロック４４０からの出力を受け、これを一時的に蓄積し、機器インタフェース１６５と互換性を持つようにデータをフォーマットする。ベースライン計測スクリプト１５４によるデータ出力も制御履歴ファイルに書き込まれる。供給された変数名に基づいて制御履歴のヘッダが生成される。ログファイルにはファイルの１行目でエンコードされたヘッダがある。計算されたヘッダがファイルの１行目と整合しない場合、既存のファイルの名前を変え、新しいものを開始する。

【００５０】

ここで図５を参照すると、本発明のもうひとつの実施形態による、単一処理ツール１２０上に複数のコントロールアクションを実装できる複数コントローラ用ベースラインプロセススクリプト５００の簡略ブロック図が示されている。たとえば、フォトリソグラフィ用ステッパにオーバーレイコントローラと臨界寸法コントローラの両方を持たせることができる。これらのコントローラは、処理済みウエハからのフィードバックを利用して、露光線量、露光時間、焦点などのさまざまなステッパパラメータを調節する。ポリシリコン層を形成するためのツールなどの成膜ツールにもポリシリコンの粒度やポリシリコン層の厚さなどのパラメータを制御するためのコントローラを複数持たせることができる。

【００５１】

ベースラインプロセススクリプト５００が呼ばれると、このスクリプトは呼出に含まれる情報に基づいて必要なコントロールアクションを判定する。ロットが処理されるコンテキストは、動作対象のコントローラがどれになるかを決める。このコンテキストは、オペレーションＩＤ、エンティティＩＤ、製品ＩＤ、個々の実行（ラン）についての要件を決める他の別個の識別子によって定義される。まず、エンティティＩＤを使ってツールのタイ

プの大雑把なクラスを求める（ステッパ、エッチャー、炉など）。たとえば、エンティティＩＤが処理ツール１２０をステッパであると識別した場合、ステッパ制御コードが呼出される。

【００５２】

ステッパ制御コードの中で、スクリプト内のコンテキスト変数がチェックされ、どの個別コントローラが呼ばれるかを判定する。オペレーションＩＤは、プロセスが走ることを示している（ポリゲートマスクｖｓ第２の層間誘電体層マスク（ＩＬＤ）など）。各コントローラはコンテキスト状況のセットに適用され、これらのコンテキスト条件がすべて満たされた場合にのみ走る。たとえば、ＣＤコントローラは、ポリゲートマスク用に走ることができるが、第２のＩＬＤマスクプロセスでは走ることはできない。一方、オーバーレイコントローラは、両方のマスクイベントで走ることができる。

【００５３】

ベースラインプロセススクリプト５００は、ツールセット（ステッパなど）に基づいて必要なツールコードを整合させるための柔軟性を提供し、利用できるコントローラ（オーバーレイ、ＣＤなど）をすべて走らせる準備をする。同じメインスクリプトを走らせ、同じサブルーチンを呼ぶことが可能であるが、現在のコンテキストで必要なコントローラのみが起動される。

【００５４】

複数コントローラ用ベースラインプロセススクリプト５００は、アプリケーションコンフィギュレーションブロック５１０と、ベースラインアプリケーションセットアップブロック５２０と、コントローラ定数およびコンテキスト固有の設定ブロック５３０と、フィードフォワードデータ分析ブロック５４０と、制御スレッドブロック５５０と、危機ブロック５６０と、コントロールアクションおよびビジネスルールブロック５７０と、結果ブロック５８０とを含む。複数コントローラ用ベースラインプロセススクリプト５００は、後述する点を除いてベースラインプロセススクリプト１５２と同じように動作する。

【００５５】

コントローラ定数およびコンテキスト固有の設定ブロック５３０は、どのコントローラを適用可能であるか（コントローラＡ、コントローラＢあるいは両方など）を判定し、過去に定義されたコンテキストおよびＲＭＳ情報を利用して、制御の動きを算出するのに各コントローラが使う値を設定する。また、コントローラ定数およびコンテキスト固有の設定ブロック５３０は、アプリケーションコンフィギュレーションブロック５１０に設定されているようなロット番号およびレイヤ名ごとのクエリを使って必要なコントローラ各々のフィードフォワード情報をデータベースから読み出す。フィードフォワードデータ分析ブロック５４０は、特定のロットに関連したデータのアレイ中のエレメントをチェックし、各コントローラについて抜けている値のデフォルト値を埋める。

【００５６】

制御スレッドブロック５５０は、データストア２３０、２３２を照会してアクティブなコントローラ各々の現在の制御スレッドに関連した制御状態を読み出すのに必要なキーと状態構造の値を設定する。これらのキーはデータストア２３０、２３２からスレッド状態データを読み出すのに使われる。制御スレッドブロック５５０は、このスレッドコンテキストで処理された新しいロットの順に並べられたデータのスタックのスレッド状態データを検索する。このような値が見つかった場合、これは制御モデルを含むユーザ定義関数に渡され、この関数でスレッド状態の値を計算して返す。スタックに値が見つからなかった場合、制御スレッドブロック３４０は階層を上がって検索し、スレッド状態の値のある最初の階層レベルからデータを読み出す。スタックおよびすべての階層レベルは同様ではあるが精度の異なるデータを含むと仮定される。

【００５７】

コントロールアクションおよびビジネスルールブロック５７０は、各コントローラの状態およびターゲット情報からコントローラ入力（プロセスレシピ更新）を計算する。使用されるコントローラは複数あるため、ひとつのコントローラがそのコントロールアクション

を判定する上で頼りにしている状態情報に他のコントローラが影響を与える場合がある。したがって、それぞれのコントロールアクションを判定する順序を決めるための相対的な優先値をコントローラに割り当てるようにしてもよい。２番目のコントローラの状態情報データを、優先度の高いコントローラがそのコントロールアクション決定に基づいて更新することができる。次に修正された状態情報に基づいて２番目のコントローラがそのコントロールアクションを判定する。このようにして連携することで、コントローラは動作レシピの変更に関して互いに競合することがなくなる。

【００５８】

結果ブロック５８０は、アクティブなコントローラすべてからコントロールアクション出力を集め、このデータを一時的に蓄積し、データをフォーマットする。結果ブロック５８０は一時的に蓄積したデータを機器インタフェース１６０に送信し、機器インタフェース１６０に対してマシンインタフェース１９５によってセットアップ／スタートマシン呼出を開始する。次に、結果ブロック５８０は、現在のコンテキスト（スレッド）についてロット番号およびレイヤに対してデータストア２３０、２３２に格納されたデータを格納し、危機スタックを更新する。

【００５９】

ここで図６を参照すると、本発明の他の実施形態による複数のコントローラを統合するための方法の簡略流れ図が示されている。ブロック６００では、複数のツールで被加工物を処理する。ブロック６１０では、（制御実行マネージャ１５０によるなどの方法で）複数のツールのうちの選択したツールについてベースライン制御スクリプトを開始する。ベースライン制御スクリプトの開始後、コントロールエグゼキュータ２２０が残りのタスクを実施する。ブロック６２０では選択したツールに必要な制御ルーチン群を特定する。ブロック６３０では、必要な制御ルーチン群について選択したツールに関連した過去のコントロールアクションに関する制御状態情報を読み出す。ブロック６４０では、必要な制御ルーチン群からの最初の制御ルーチンを実行し、最初のコントロールアクションを生成する。ブロック６５０では、最初のコントロールアクションに基づいて必要な制御ルーチン群からの２番目の制御ルーチンに関連した制御状態情報を変更する。ブロック６６０では、修正後の制御状態情報に基づいて２番目の制御ルーチンを実行し、２番目のコントロールアクションを生成する。

【００６０】

本発明は本願明細書の教示内容の利益を享受する当業者らに自明の上記とは異なるが等価な方法で改変および実施することのできるものであるため、上記にて開示した個々の実施形態は一例にすぎない。さらに、添付の請求の範囲に記載したものを除き、本願明細書に図示した構造または設計の詳細に対する限定を何ら意図するものではない。したがって、上記にて開示した個々の実施形態を変更または改変してもよく、そのような変形例はいずれも本発明の範囲に包含されるとみなせることは明白である。よって、本願明細書で求める保護は添付の請求の範囲に記載のとおりである。

【図面の簡単な説明】

【００６１】

【図１】本発明の一実施形態による高度プロセス制御（ＡＰＣ）システムの簡略ブロック図である。

【図２】図１のシステム内のベースライン制御スクリプトと、多様な共有ベースラインライブラリとの間のリンクを示す図である。

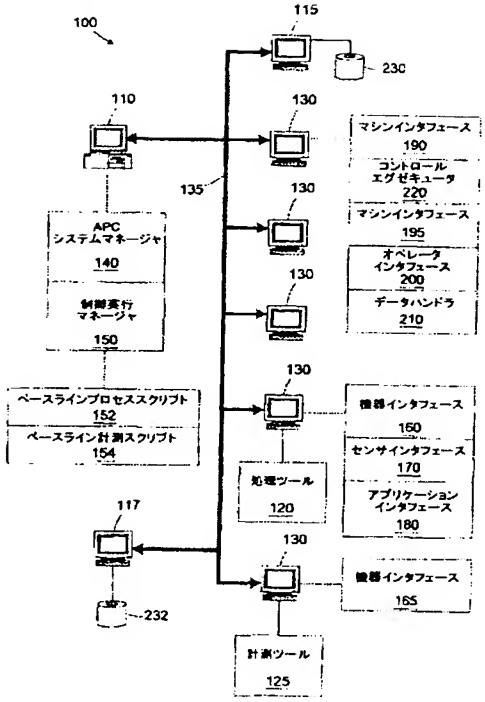
【図３】ベースラインプロセススクリプトの構成を示す簡略ブロック図である。

【図４】ベースライン計測スクリプトの構成を示す簡略ブロック図である。

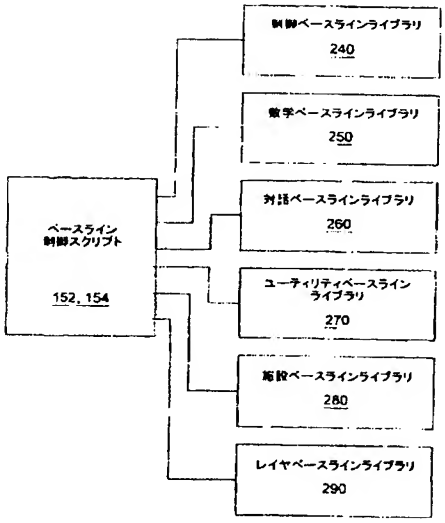
【図５】複数コントローラ用ベースラインプロセススクリプトの構成を示す簡略ブロック図である。

【図６】本発明のもうひとつの実施形態による複数のコントローラを統合するための方法の簡略流れ図である。

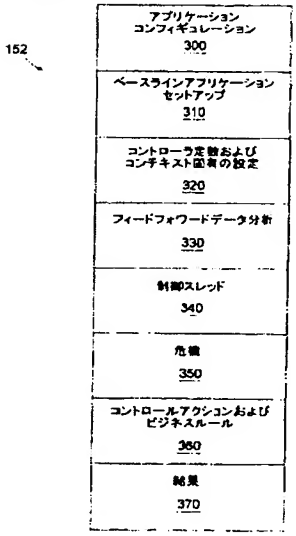
【図１】


100
115
230
110
130
135
マシンインタフェース
190
コントロール
エグゼキュータ
220
マシンインタフェース
195
オペレータ
インタフェース
200
データハンドラ
210
APC
システムマネージャ
140
制御実行
マネージャ
150
ベースラインプロセススクリプト
152
ベースライン計測スクリプト
154
機器インタフェース
160
センサインタフェース
170
アプリケーション
インタフェース
180
処理ツール
120
117
232
機器インタフェース
165
計測ツール
125

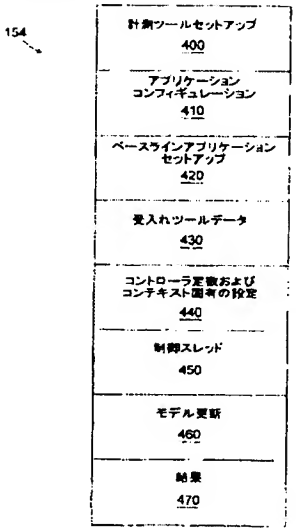
【図２】


制御ベースラインライブラリ
240
数学ベースラインライブラリ
250
対話ベースラインライブラリ
260
ユーティリティベースライン
ライブラリ
270
施設ベースラインライブラリ
280
レイヤベースラインライブラリ
290
ベースライン
制御スクリプト
152, 154

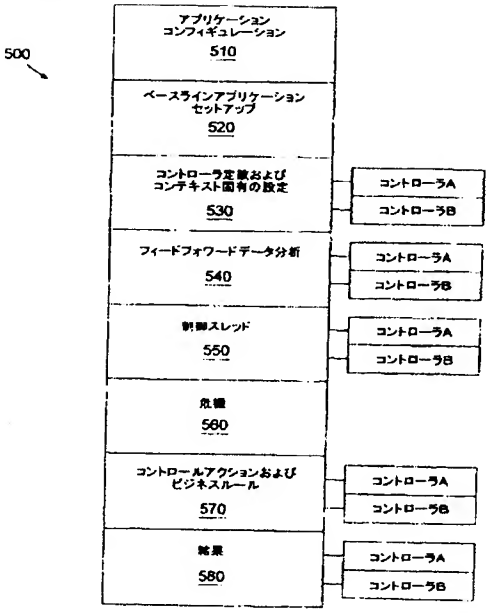
【図３】


152
アプリケーション
コンフィギュレーション
300
ベースラインアプリケーション
セットアップ
310
コントローラ定数および
コンテキスト固有の設定
320
フィードフォワードデータ分析
330
制御スレッド
340
危機
350
コントロールアクションおよび
ビジネスルール
360
結果
370

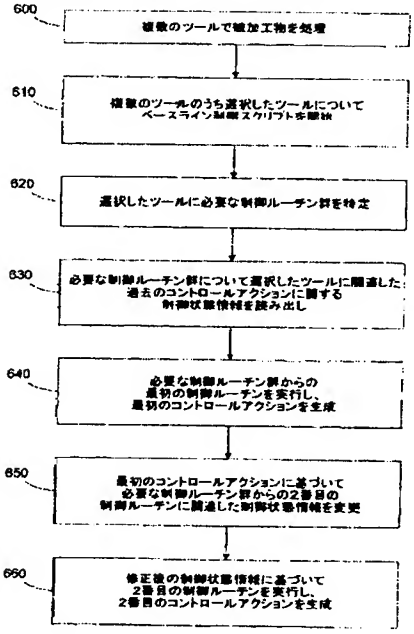
【図４】


154
計測ツールセットアップ
400
アプリケーション
コンフィギュレーション
410
ベースラインアプリケーション
セットアップ
420
受入れツールデータ
430
コントローラ定数および
コンテキスト固有の設定
440
制御スレッド
450
モデル更新
460
結果
470

【図５】


500
アプリケーション
コンフィギュレーション
510
ベースラインアプリケーション
セットアップ
520
コントローラ定数および
コンテキスト固有の設定
530
フィードフォワードデータ分析
540
制御スレッド
550
危機
560
コントロールアクションおよび
ビジネスルール
570
結果
580
コントローラA
コントローラB
コントローラA
コントローラB
コントローラA
コントローラB
コントローラA
コントローラB
コントローラA
コントローラB

【図６】


600
複数のツールで被加工物を処理
610
複数のツールのうち選択したツールについて
ベースライン制御スクリプトを開始
620
選択したツールに必要な制御ルーチン群を特定
630
必要な制御ルーチン群について選択したツールに関連した
過去のコントロールアクションに関する
制御状態情報を読み出し
640
必要な制御ルーチン群からの
最初の制御ルーチンを実行し、
最初のコントロールアクションを生成
650
最初のコントロールアクションに基づいて
必要な制御ルーチン群からの2番目の
制御ルーチンに関連した制御状態情報を変更
660
修正後の制御状態情報に基づいて
2番目の制御ルーチンを実行し、
2番目のコントロールアクションを生成